School of Computer Science

60-265-01 Computer Architecture and Digital Design

Winter 2009

Midterm Examination # 2

Wednesday, March 18, 2009

Student Name: ________________ ____________________
First Name Family Name

Student ID Number: _____________________

Duration of examination: 75 minutes

1. Answer all questions on this examination paper in the space provided.
2. This is a closed-book examination – no notes or books or electronic computing or storage devices may be used.
3. Do not copy from other students or communicate in any way. All questions will be answered only by the attending proctors.
4. All students must remain seated during the last 5 minutes of the examination.
5. The examination must be surrendered immediately when the instructor announces the end of the test period.
6. Each student must sign the examination list before leaving the classroom.

Total mark obtained: __________________

Maximum mark: 41

Question 1. [10 marks]
Answer all parts of this question.

A. State briefly what is wrong with the following register transfer statement. [1 mark]

```
Q  :  R1 = R2 , R1 = R3
```

Buses cannot be enabled to permit two different sources of data to be transferred to a single location at the same time.

B. State briefly what is wrong with the following register transfer statement. [1 mark]

```
Q  :  PC = AR , PC = PC + 1
```

You cannot enable both the LOAD and INC inputs on the PC register at the same time.

C. Register R1 contains the 8-bit binary value 11011001. Determine the binary value that must be in a register R2, plus determine the logic operation ***op*** to be performed, such that the value in R1 is changed to 01101101 after the operation is performed. [2 marks]

R1 = R1 op R2 11011001 XOR 10110100 = 01101101

Op is XOR, R2 = 10110100 (Check for alternatives)

D. Register R1 contains the 8-bit binary value 11011001. Determine the binary value that must be in a register R2, plus determine the logic operation ***op*** to be performed, such that the value in R1 is changed to 11111101 after the operation is performed. [2 marks]

R1 = R1 op R2 11011001 XOR 00100100 = 11111101

Op is XOR, R2 = 00100100 (Check for alternatives)

E. Starting from an initial value of R = 11011101, determine the sequence of binary values in R after a logical shift-left, followed by a circular shift-right, followed by a logical shift-right, and followed, finally, by a circular shift-left. Show all your work. [4 marks]

Lshl(11011101) leads to 10111010
Cshr(10111010) leads to 01011101
Lshr(01011101) leads to 00101110
Cshl(00101110) leads to 01011100

Question 2. [11 marks]

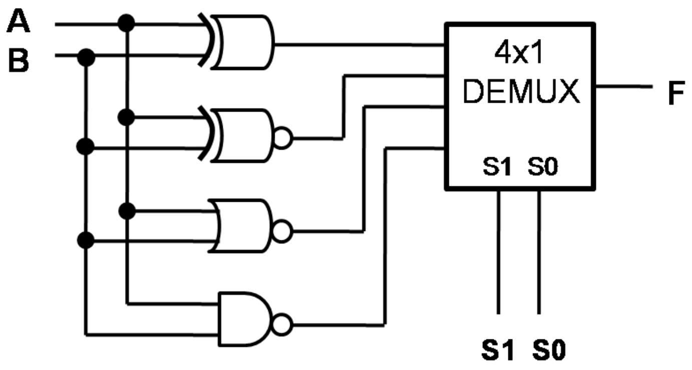
A. State how many 128 x 8 memory chips (number of addresses times number of bits at each address) are needed to provide a memory capacity of 4096 x 16. **[2 marks]**

Break into two sub-questions: How many 128's go into 4096? Answer: 32
How many 8's go into 16? Answer: 2
Thus, we need 32x2=64 memory chips.

Added comment:
In most cases of manufacturing, these chips are aligned in tabular arrangements. Noting that two chips are needed to define the 16-bit storage at a given address (that spans both chips), access to each chip requires both decoders and MUXes and enable logic.

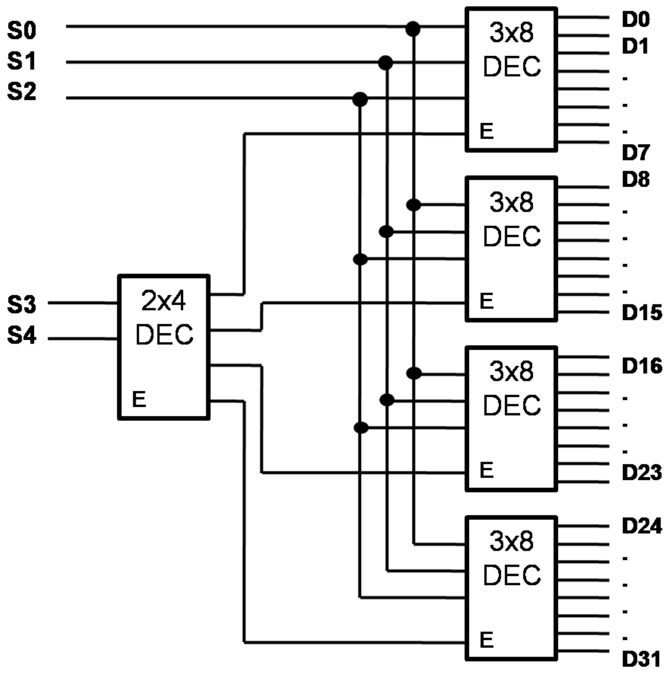
B. Design a digital circuit that performs the four logic operations of exclusive-OR, exclusive-NOR, NOR and NAND. Use two selection variables. Show the block diagram assuming two 1-bit inputs A and B. The output from the circuit is F. **[5 marks]**

The circuit block diagram is shown below. Note how each logic gate is applied to A and B, in parallel, but the MUX serves to select one of these to pass through to the output F.



C. Construct a **5-to-32** line decoder with four **3-to-8** decoders, with enable, and one **2-to-4** line decoder, with enable. Use block diagrams. Number the inputs as S_0, S_1, up to S_5. Similarly, label the outputs as D_0, D_1, up to D_{31}. You may use abbreviations in your answer (ie. you do not need to state every line and every label). **[4 marks]**

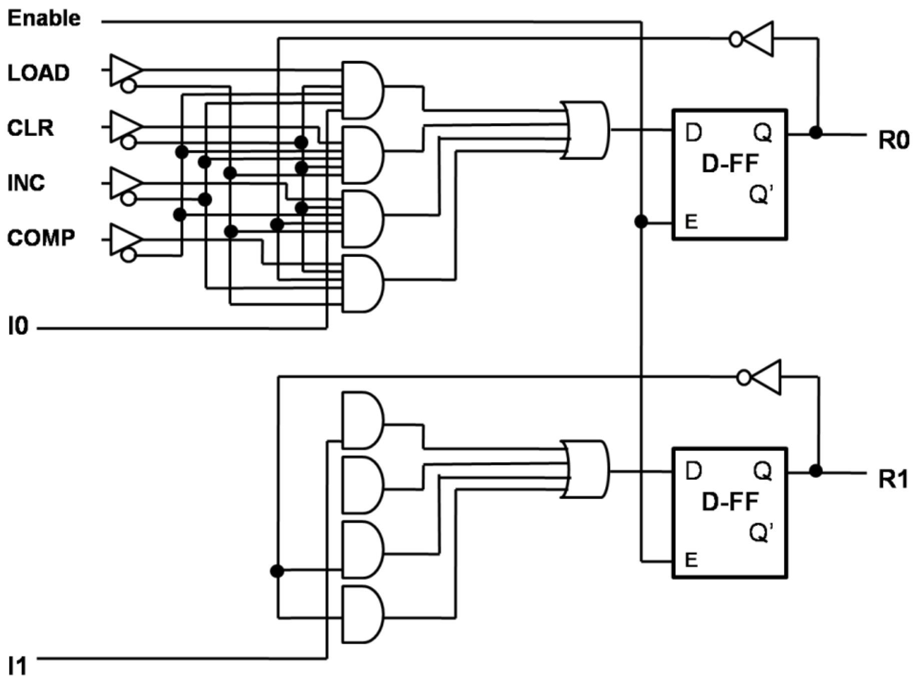
A similar problem was posed in previous exercises – check your notes and website sources. The circuit is shown below. Note how the outputs from the 2-to4 DEC are attached to the Enable inputs on the 3-to8 DECs.



Question 3. [10 marks]
Design and draw the block logic diagram for a digital circuit for a 2-bit register R, with LOAD, CLEAR, INCREMENT and COMPLEMENT enable inputs. You may use any flip-flop that you wish to represent each bit of R. Show all other elements of the circuit logic. You must make sure that all block elements used are properly labeled and identified.

This is answered in various ways. The one shown below is based on an example developed in the lecture notes (and in Mano). I have used a D-flip-flop, hence I must always assert the loading value, but I depend on the fact that the D-ff is Enabled, so that when E=0 a refresh automatically happens, while when E=1, another of the input options is performed.

The circuit provided below is not complete. I have shown all details for the R0 bit, but I have left the handling of the R1 bit as an exercise. Most of the circuit for R1 is similar or identical to that of R0, since LOAD, CLEAR and COMPLEMENT are the same. Only for INCREMENT, must an additional interface logic circuit be inserted, namely, the circuit (R0 XOR R1) which is obtained from truth-tables.



Question 4. [10 marks]

A. Using register transfer language, state precisely what control conditions and microoperations are required to fully specify the complete instruction cycle for the instruction whose machine code is the 16-bit value, stated in hexadecimal: **`B2E4`**. **[5 marks]**

From the attached page, obtain the opcode from the leading hex digit, B or binary 1011. This implies that the I-bit of the instruction is 1, for indirect addressing, and the opcode is STA, or Store the Accumulator to memory. This is what we need in order to express the full instruction cycle.

Using the timing logic supplied by the Sequence Counter register and Decoder, together with the I-bit to specify the addressing mode and the instruction opcode decoder output D3 (STA=3), we have the following control logic and microoperations:

```
        T0 : AR = PC , SC = SC + 1
        T1 : IR = M[AR] , PC = PC + 1 , SC = SC + 1
        T2 : { D7, .. D3, .. D0 } = DEC( IR(12-14) ) ,
             AR = IR(0-11) , I = IR(15), SC = SC + 1
  D3.I.T3 : DR = M[AR] , SC = SC + 1
  D3.I.T4 : AR = DR(0-11) , SC = SC + 1
  D3.I.T5 : AC = M[AR] , SC = 0
```

Recalling from the lecture discussion that Mano's description is vague on the use of the DR register (ie. how is the data transferred from the memory, M, to either IR, or AC, or AR, etc), I have included it above with some liberties. I have tried to be consistent with Mano's description for steps 0, 1, 2 and 5, but I have referred to DR in steps 3 and 4.

B. Using register transfer language, state precisely what control conditions and microoperations are required to fully specify the complete instruction cycle for the instruction whose machine code is the 16-bit value, stated in hexadecimal: `6320`. **[5 marks]**

From the attached page, obtain the opcode from the leading hex digit, 6 or binary 0110. This implies that the I-bit of the instruction is 0, for direct addressing, and the opcode is ISZ, or Increment and Skip if Zero. This is what we need in order to express the full instruction cycle.

Using the timing logic supplied by the Sequence Counter register and Decoder, together with the I-bit to specify the addressing mode and the instruction opcode decoder output D6 (ISZ=6), we have the following control logic and microoperations:

```
            T0 :  AR = PC , SC = SC + 1
            T1 :  IR = M[AR] , PC = PC + 1 , SC = SC + 1
            T2 :  { D7, D6, ... D0 } = DEC( IR(12-14) ) ,
                  AR = IR(0-11) , I = IR(15), SC = SC + 1
      D6.I'.T3 :  AC = M[AR] , SC = SC + 1
      D6.I'.T4 :  AC = AC + 1 , SC = SC + 1
      D6.I'.T5 :  M[AR] = AC , SC = SC + 1
(AC)'.D6.I'.T6 :  PC = PC + 1 , SC = 0
 (AC).D6.I'.T6 :  SC = 0
```

Check over this logic carefully – ask WHY? is it necessary to do it this way. (Note that it still permits some alternatives, including the matter of how to include the DR register).

This page contains various definitions and information provided freely for each student to use for the examination, if required. You may detach this page.

Operation Codes and Mnemonics for Mano's machine:

Memory Access Opcodes (4 bits) – Direct (Indirect)

0 (8)	AND	1 (9)	ADD	2 (A)	LDA	3 (B)	STA
4 (C)	BUN	5 (D)	BSA	6 (E)	ISZ		

CPU based Opcodes (16 bits)

7800	CLA	7400	CLE	7200	CMA	7100	CME
7080	CIR	7040	CIL	7020	INC	7010	SPA
7008	SNA	7004	SZA	7002	SZE	7001	HLT

I/O based Opcodes (16 bits)

F800	INP	F400	OUT	F200	SKI	F100	SKO
F080	ION	F040	IOF				

Boolean Postulates:

P0: Existence: There exist at least two elements x,y in B such that x ≠ y
P1: Closure: For every x,y in B there exist two combinational operators + and . where x+y is in B and x.y is in B
P2: Identity: There exist identity elements 0,1 in B relative to the operations + and ., such that for every x in B: 0+x = x+0 = x and 1.x = x.1 = x.
P3: Commutativity: The operations + and . are commutative for all x,y in B: x+y = y+x and x.y = y.x
P4: Distributivity: Each operation + and . is distributive over the other; that is, for all x,y,z in B: x.(y+z) = x.y+x.Z and x+(y.z) = (x+y).(x+z)
P5: Complementation: For every element x in B there exists an element ~x, called the complement of x, satisfying: x+~x = 1 and x.~x = 0

Fundamental Logic Micro-operations:

F = 0	F = AB	F = AB′	F = A
F = A′B	F = B	F = AB′+A′B	F = A+B
F = A′B′	F = AB+A′B′	F = B′	F = A+B′
F = A′	F = A′+B	F = A′+B′	F = 1